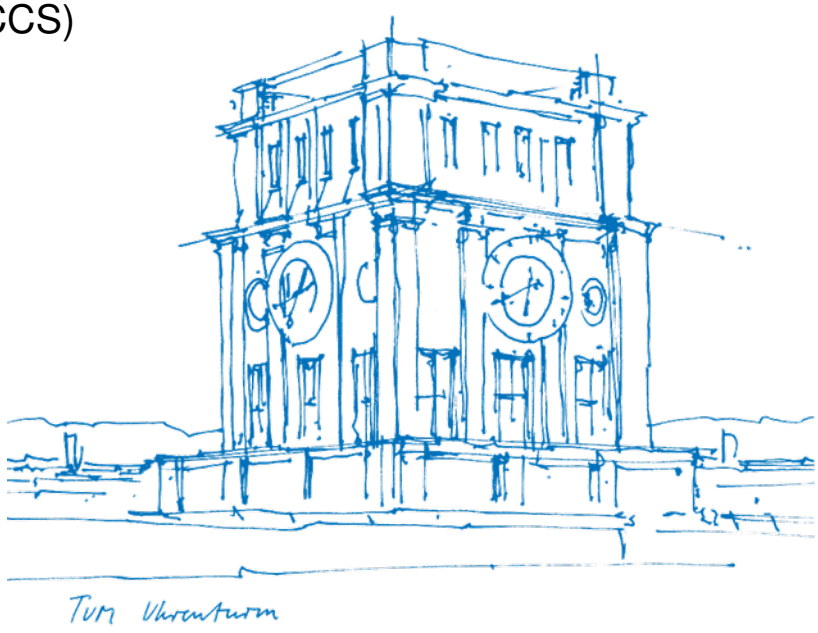
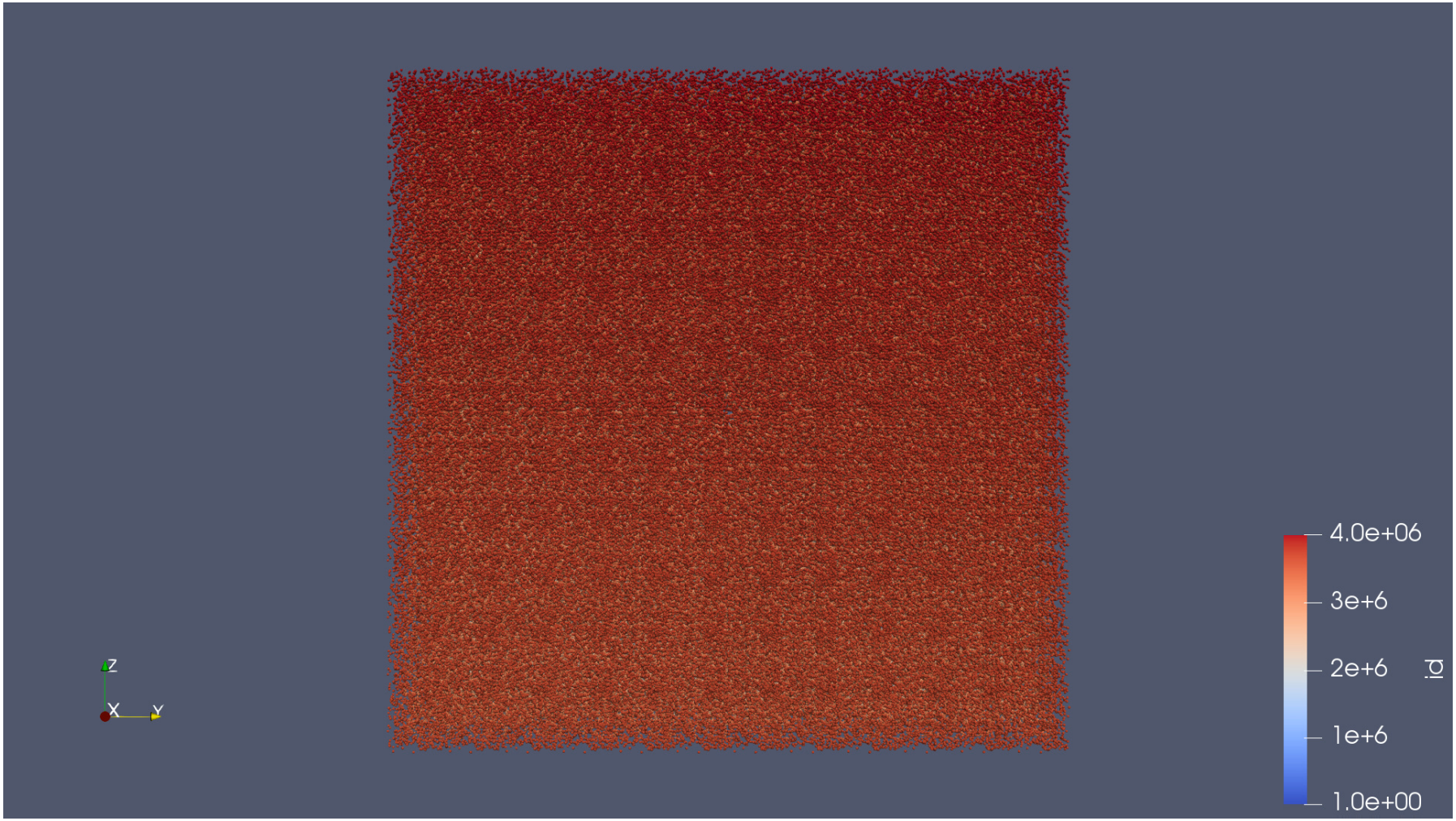
AutoPas: Auto-Tuning for Particle Simulations

Fabio Gratl
Technical University of Munich
Faculty of Informatics
Chair of Scientific Computing in Computer Science (SCCS)
Rio de Janeiro, May 24. 2019



Bundesministerium
für Bildung
und Forschung

Motivation



Outline

Molecular Dynamics

- Basics
- Challanges

AutoPas

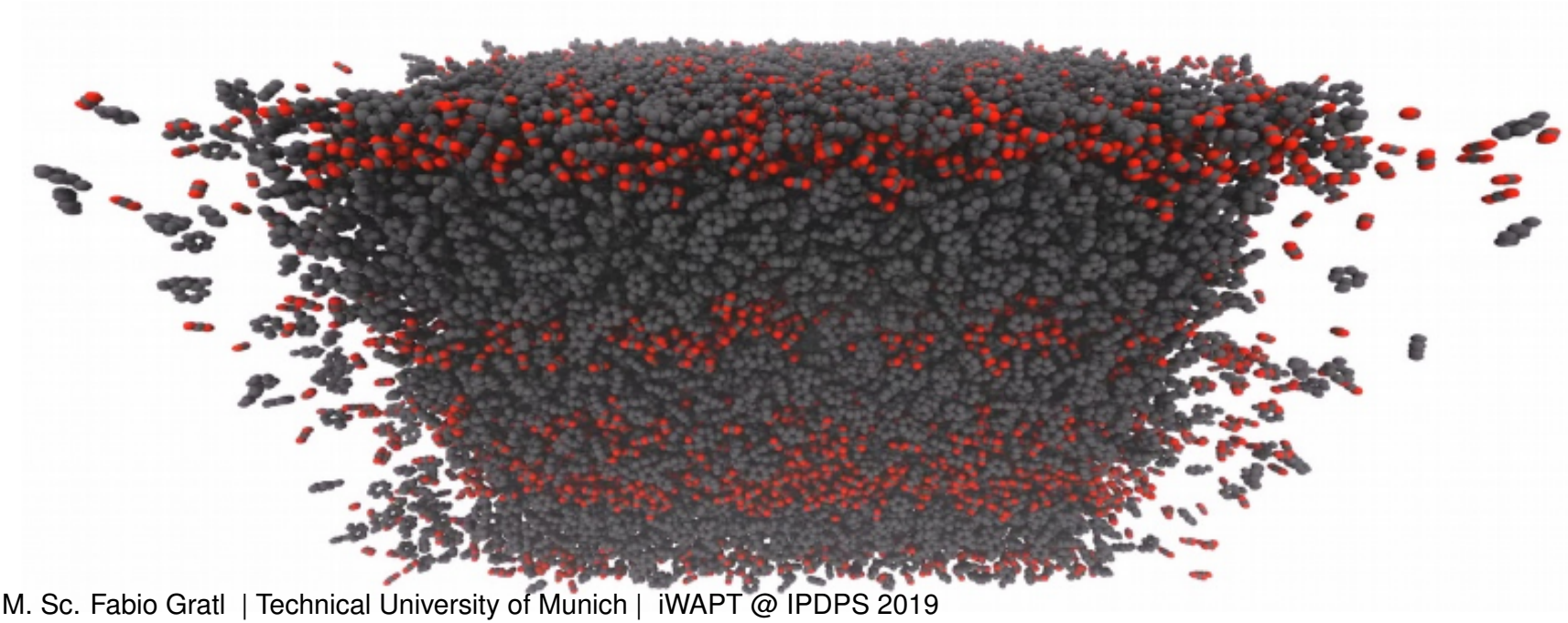
- Implemented Algorithms
- Auto-Tuning
- Integration

Experimental Results

Molecular Dynamics

Molecular Dynamics

- Here: small rigid molecules
- Simulation of movement of molecules
- Computation of pairwise forces
- Newtons Laws of Motion
- N-Body problem $\Rightarrow O(N^2)$



Short Range Interactions: Lennard-Jones Potential

$$U(x_i, x_j) = 4\varepsilon \left(\left(\frac{\sigma}{||x_i - x_j||_2} \right)^{12} - \left(\frac{\sigma}{||x_i - x_j||_2} \right)^{6} \right) \tag{1}$$

- Characteristics of molecule type:
  - ε : Potential well
  - σ : Zero crossing
- Cutoff r_c

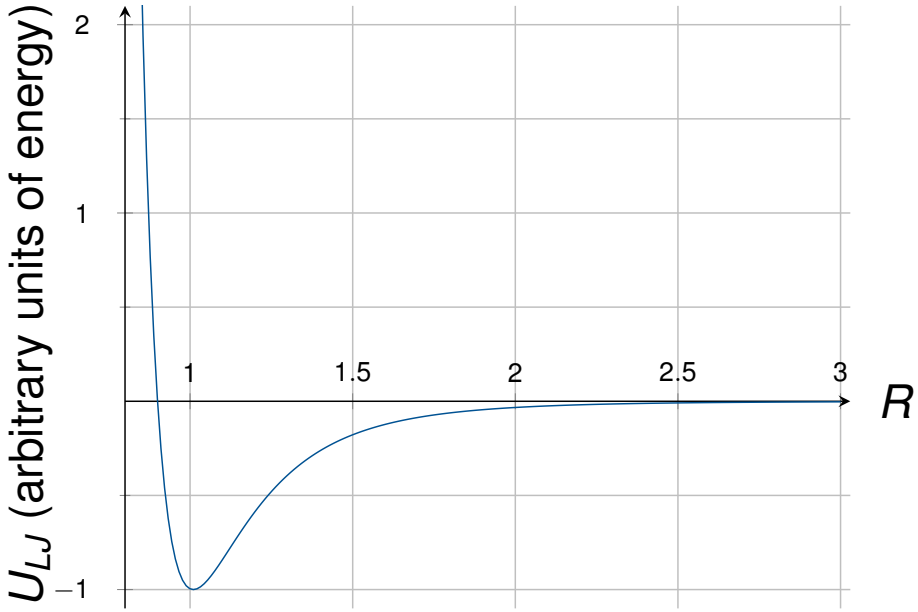
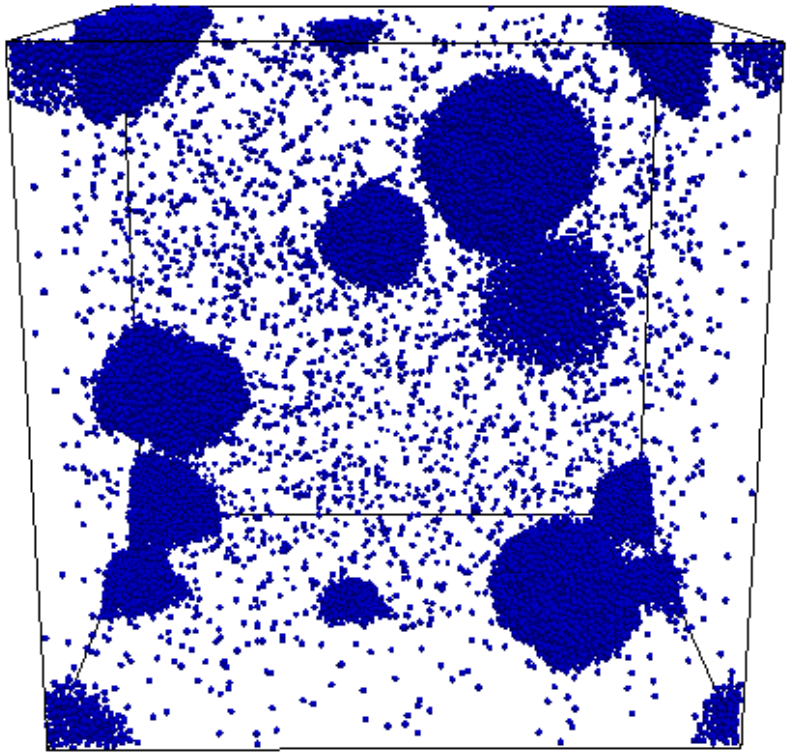


Figure: LJ-Potential for $\varepsilon = 1$ and $\sigma = 0.9$

Challanges

- Total number of particles
- Particle density
- (In-)Homogeneity
- Systems changing over time
- Many possible algorithms
- Overall goal:
  Minimize time to solution!

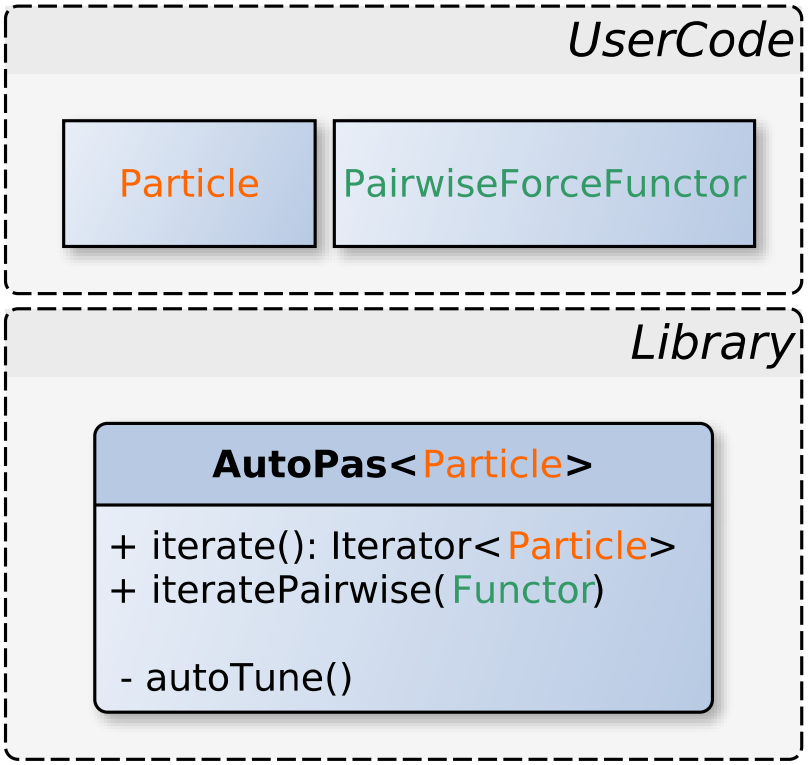


AutoPas

AutoPas: Overview

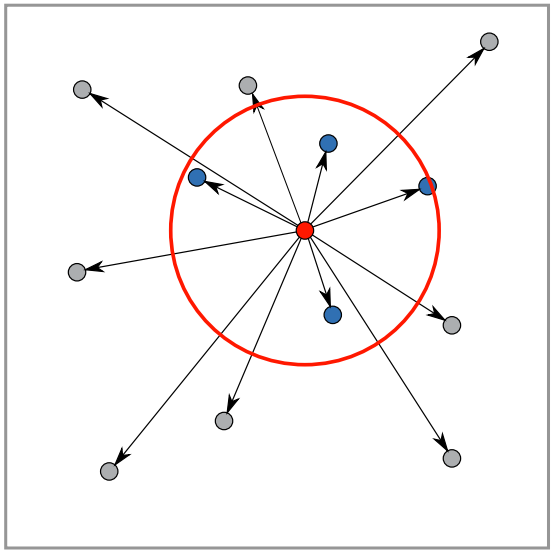
- Node-Level C++ header library
- User defines:
  - Properties of particles
  - Force for pairwise interaction
- AutoPas provides:
  - Containers, Traversals, Data Layouts, ...
  - Dynamic Tuning at run-time
- Black Box container

$\Rightarrow$ General base for *N*-Body simulations

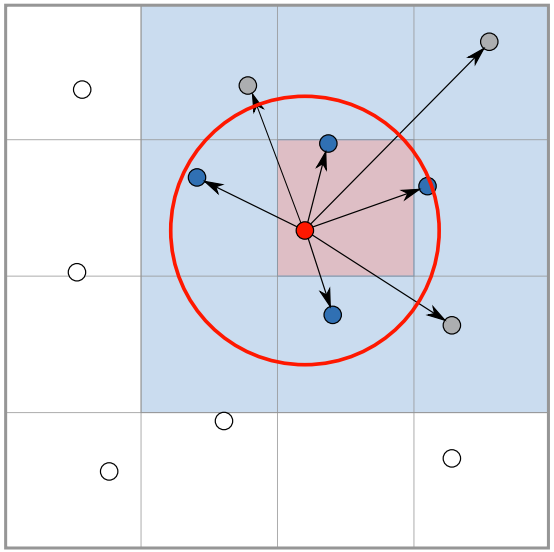


https://github.com/AutoPas/AutoPas

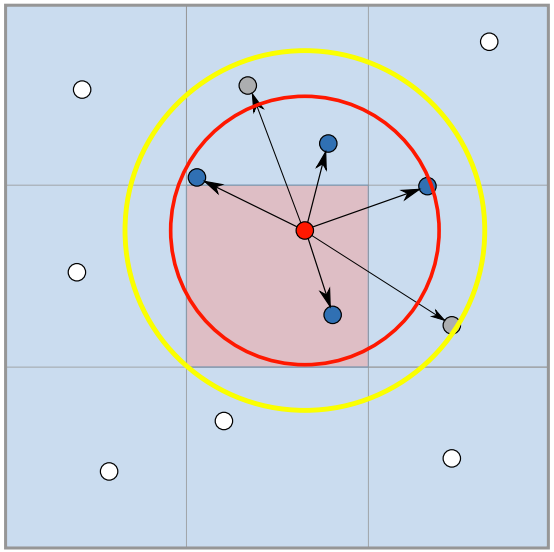
Container Options



Direct Sum



Linked Cells



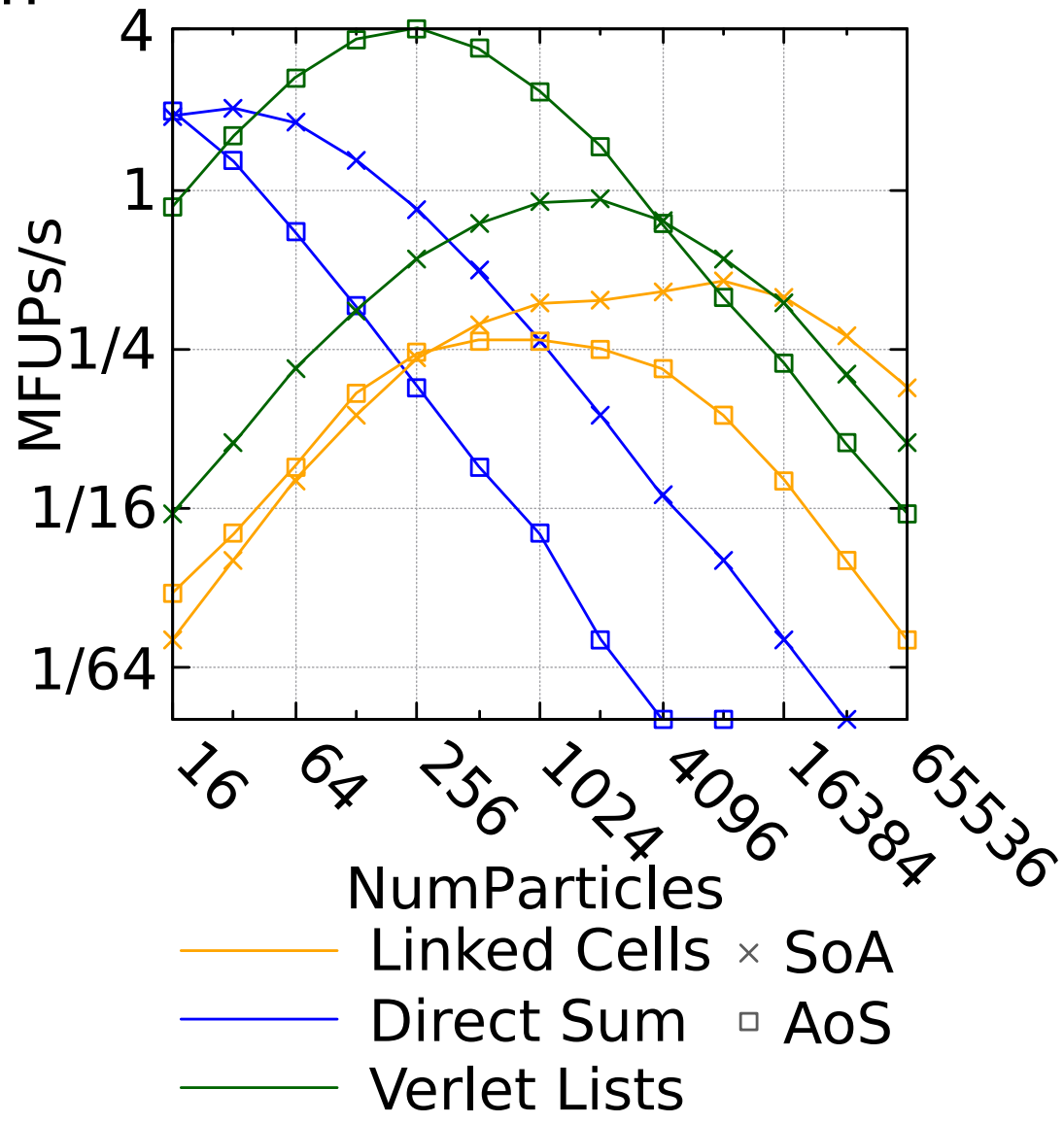
Verlet Lists

Memory Overhead

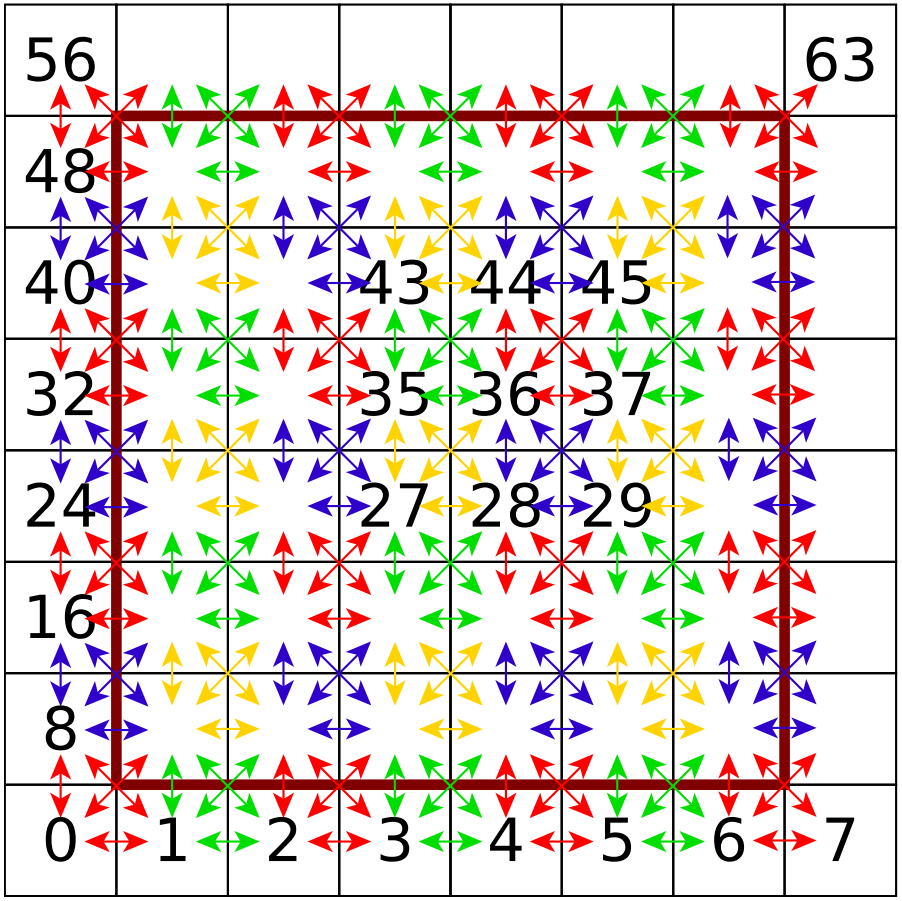
Computational Overhead

Container Comparison

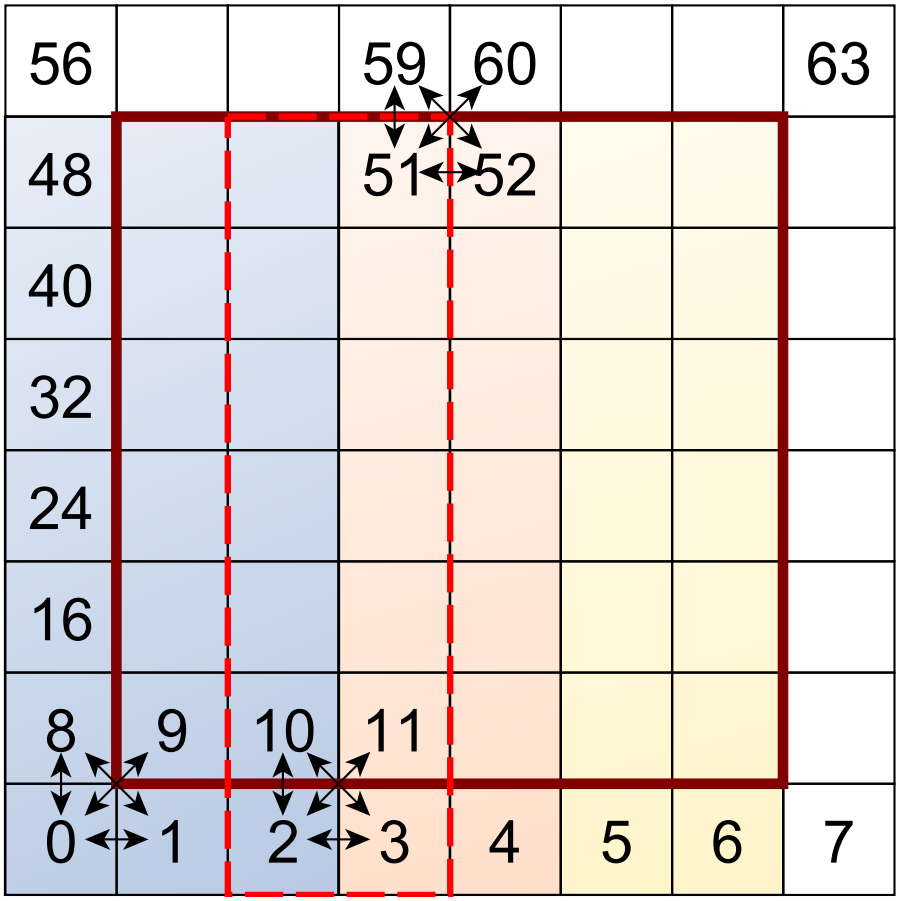
- Every container has advantages
- Linked Cell benefits most from SoA
  $\Rightarrow$best in dense scenarios



Linked Cells Parallelization Options



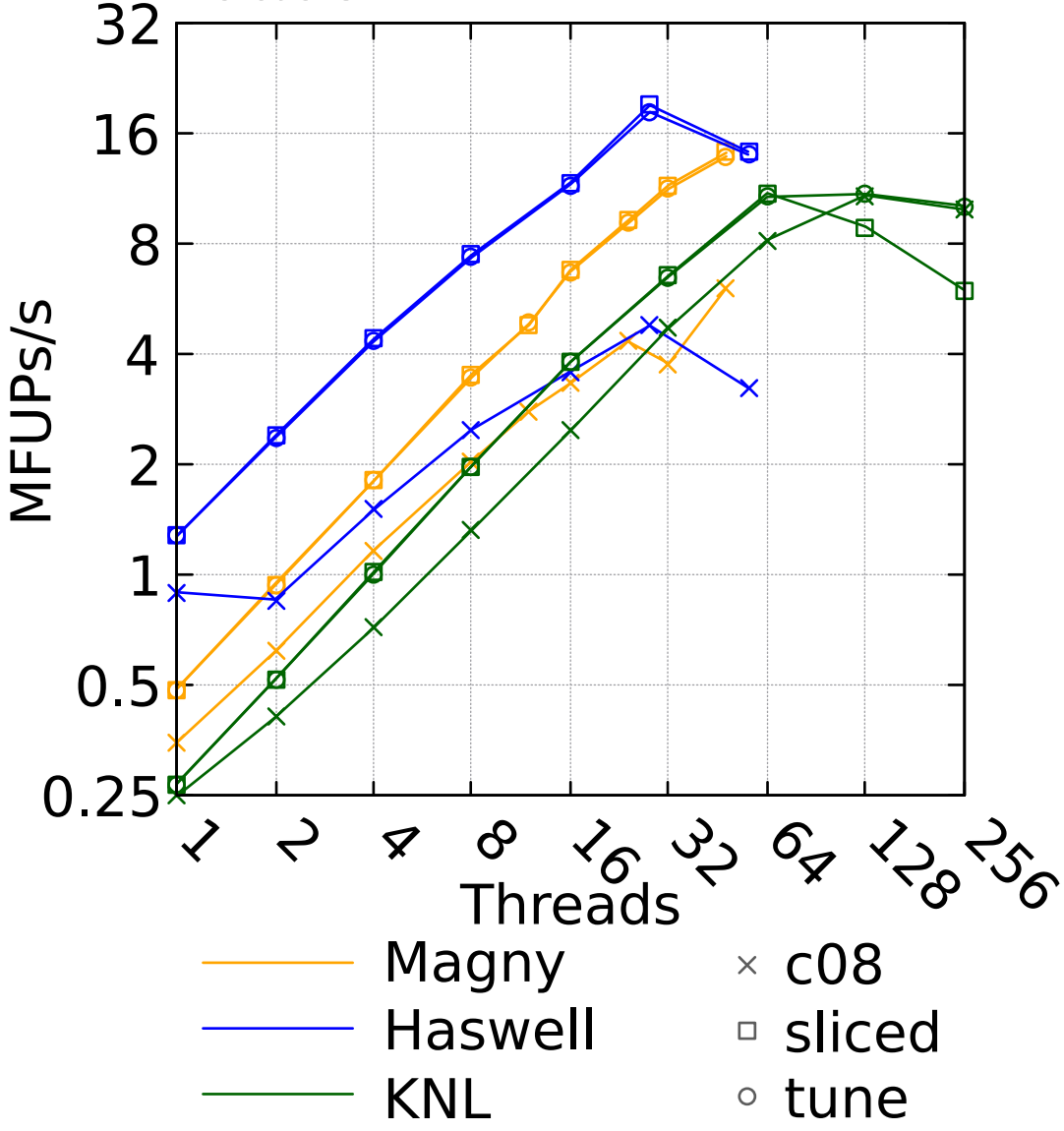
c08



sliced

Hardware Comparison / Threads

- Traversals:
  c08: 8-way domain coloring
  sliced: regular 1D domain partitioning

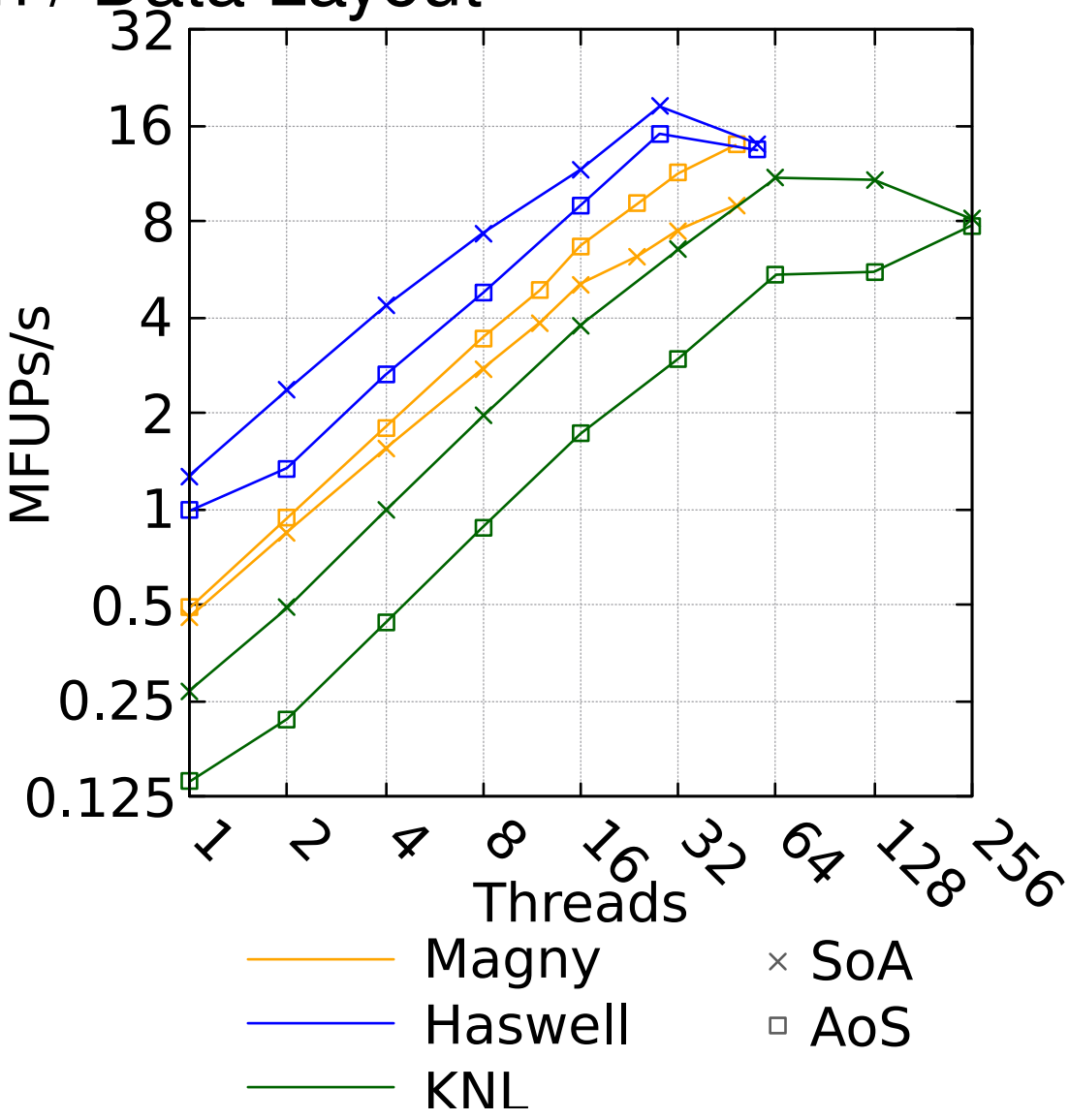


Hardware Comparison / Data Layout

- Vector Instructions:

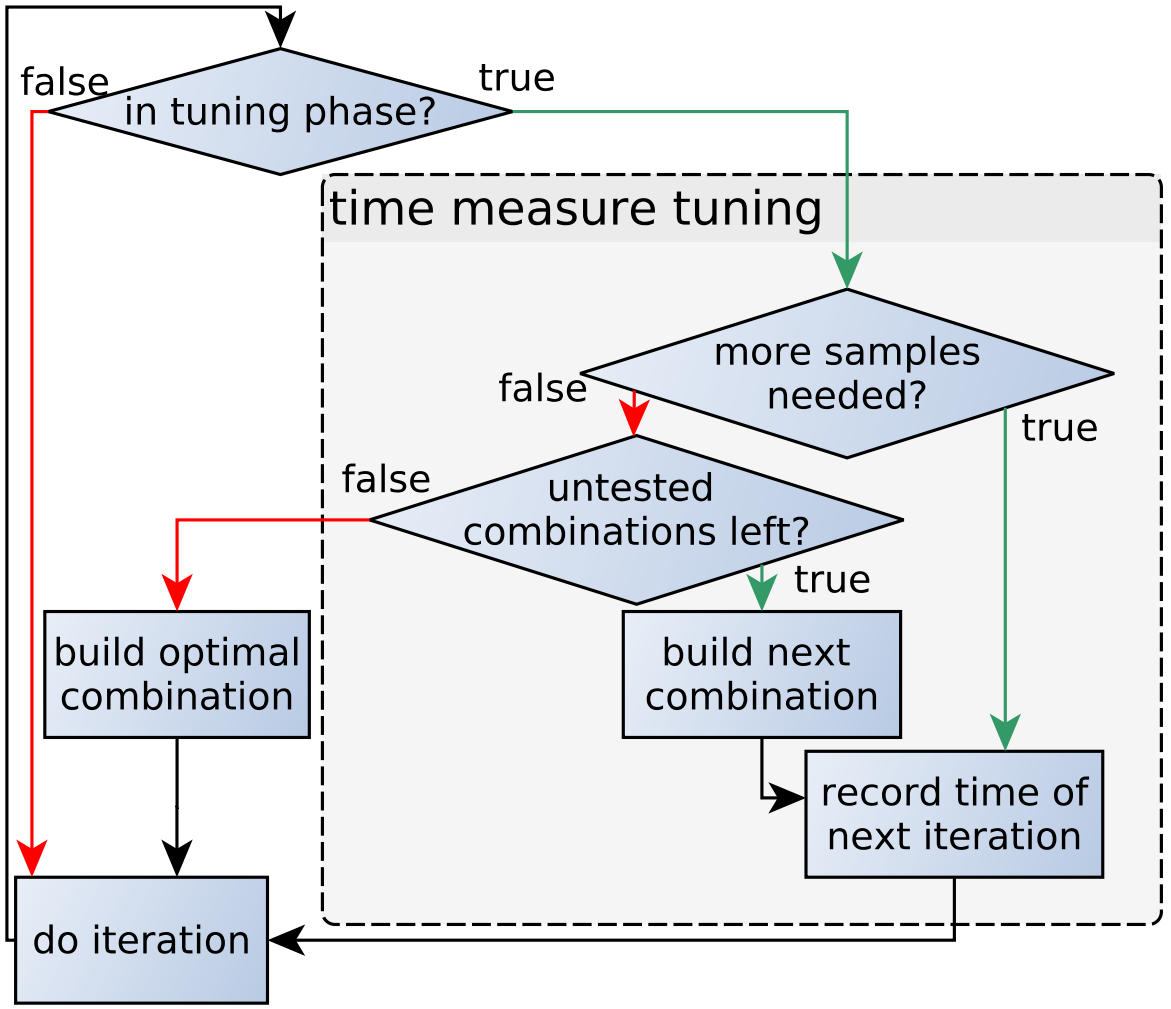
Magny	SSE4	128
Haswell	AVX2	256
KNL	AVX512	512

⇒ Optimal data layout dependent on hardware



Auto-Tuning Process

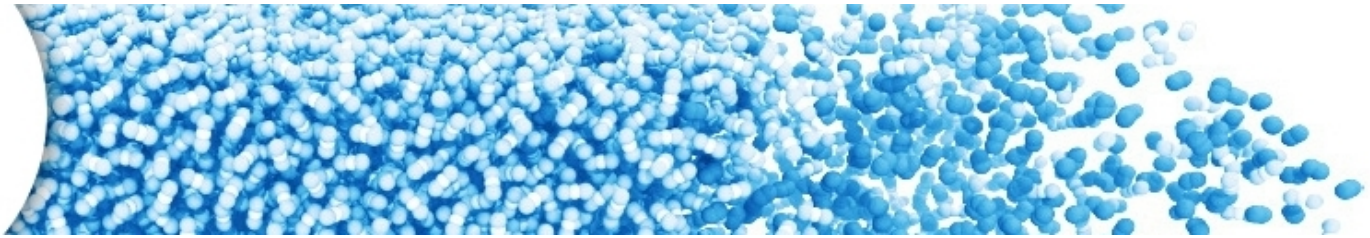
- Common interfaces for containers, traversals, etc $\Rightarrow$ Strategy pattern
- Repeated periodically
- User can restrict testing space



Integration into ls1 mardyn

- ls1 mardyn:
  - Large number of small rigid molecules.
  - Actively used in chemical engineering.
- Example Lennard-Jones functor from AutoPas
- New particle class
  - Inherits from AutoPas and ls1 mardyn particle interface.
  - Acts as coupler
- New particle container class
  - Only wrapper around AutoPas main interface.

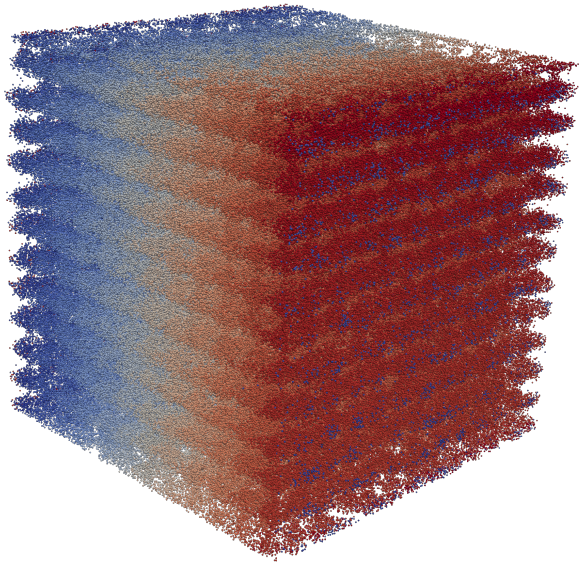
ls1 Mardyn

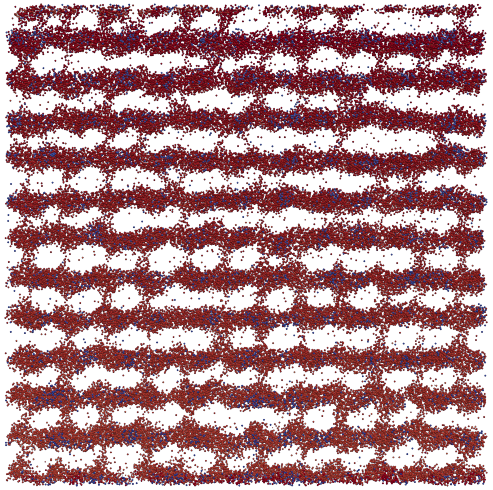


Experimental Results

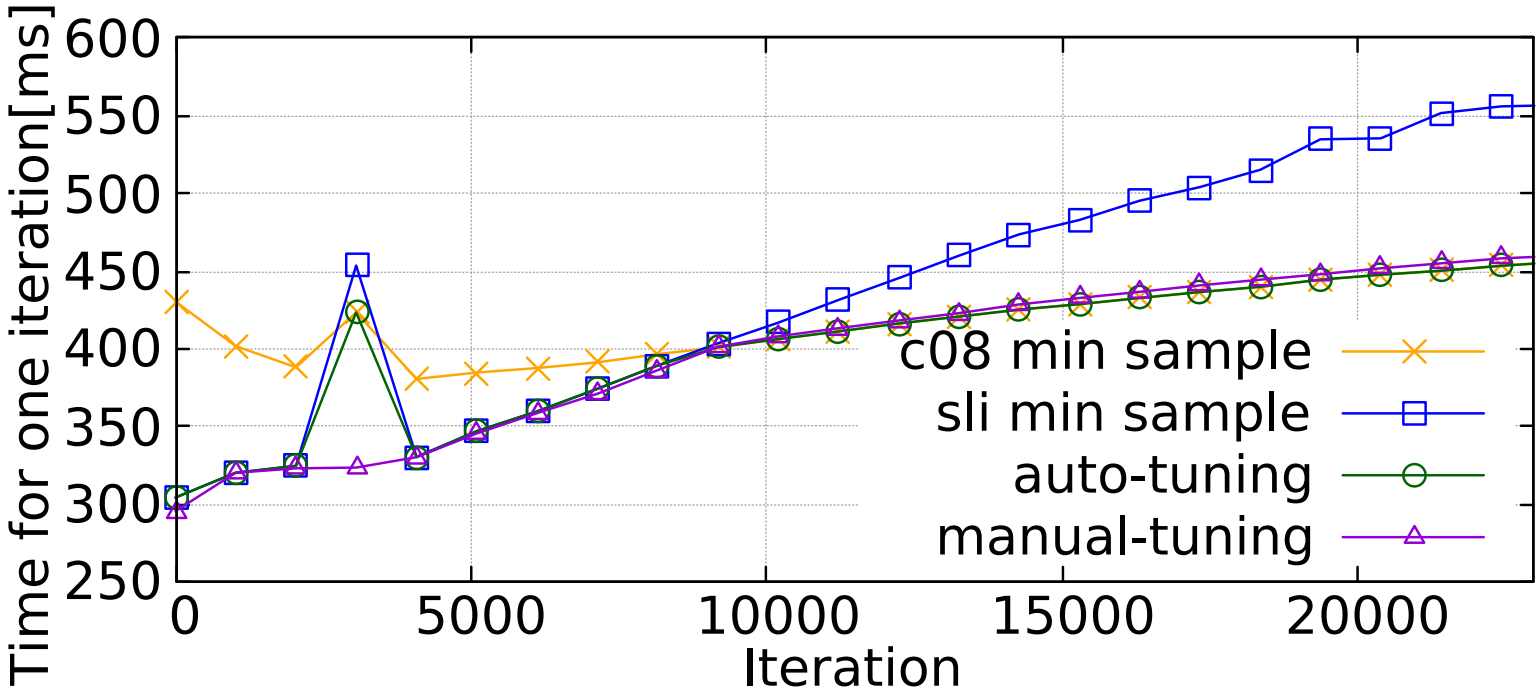
Scenario: Spinodal Decomposition

- 4 008 960 particles
- Block size: $240 \times 240 \times 240$
- Periodic boundary conditions
- Cutoff = 2.5
- Sub-critical temperature
- Rapid and drastic change in homogeneity
  $\Rightarrow$ Interesting target for tuning!



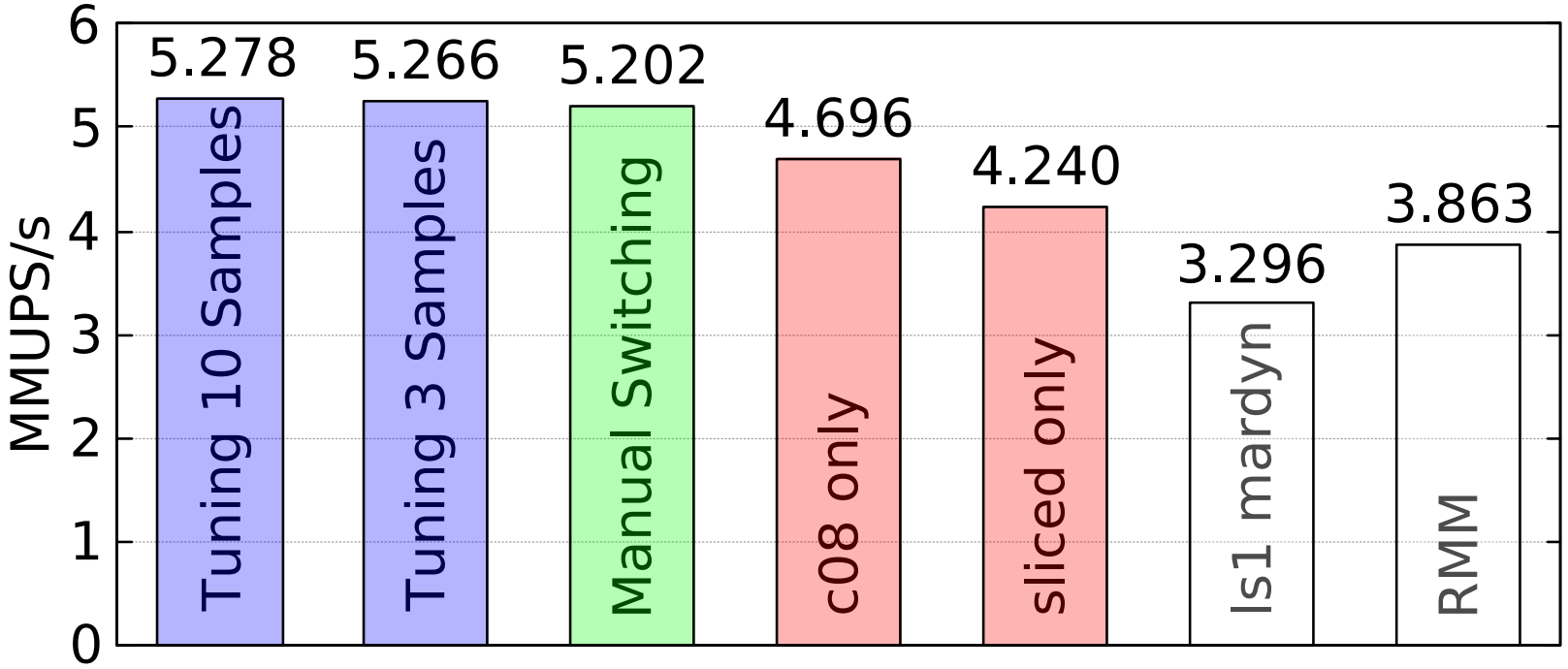


Tuning Behavior



- Tuning switches as expected
- Misclassification can happen

Tuning Overhead



- Tuning and manual switching equally fast.
  $\Rightarrow$ No overhead from tuning!
- Tuning faster than static configuration.
- Faster than original ls1 mardyn, even in Reduced Memory Mode.

Conclusions

- AutoPas is a black box *N*-Body container.
- Dynamic tuning enables optimal performance for changing scenarios.
- Achievable for users without expert knowledge.
- Easy to integrate in existing codes.

... and future work:

- AutoPas + MPI.
- More / optimized algorithms.
- Tuning for more parameters.
- Search space reduction.